Security Assessment Methodology for Mobile Applications

Álvaro Botas[†], Ricardo J. Rodríguez[†], Jesús Balsa[†], Juan Felipe García[†], Javier Alonso[†], Francisco J. Lera[‡], Christian García[‡], Vicente Matellán[‡], and Raúl Riesco[♯]

[†]Research Institute of Applied Sciences in Cybersecurity, Universidad de León, Spain
[‡]Escuela de Ingenierías Industrial e Informática, Universidad de León, Spain
[♯]Spanish National Institute of Cybersecurity (INCIBE)

Abstract

Any type of software, from desktop to mobile applications, is prone to contain defects that can lead to vulnerabilities. These vulnerabilities, when exploited, may put in risk the integrity, confidentiality and availability of the software. Security auditing methodologies help to reduce at some level of confidence these risks. With the explosion of mobile applications for daily activities like checking email, news, social networks, or even managing bank accounts, guaranteeing an acceptable level of application security becomes critical for the usage and trust of mobile services. In this paper, we review and classify OWASP 2014 Top Ten mobile risks in analysis blocks. Based on the blocks classification, we propose a methodology to security audit mobile software applications. We demonstrate the effectiveness of the proposed methodology by auditing the same mobile application in Google's Android and Apple's iOS platforms surfacing multiple vulnerabilities.

Analysis Blocks to Identify Mobile Risks

OWASP 2014 Top Ten Mobile Risks [4]

(M1) Weak Server Side Controls
(M2) Insecure Data Storage
(M3) Insufficient Transport Layer Protection
(M4) Unintended Data Leakage
(M5) Poor Authorization and Authentication
(M5) Broken Cryptography
(M6) Client Side Injection
(M7) Security Decisions via Untrusted Inputs
(M8) Improper Session Handling
(M9) Lack of Binary Protection

Analysis blocks proposed

- **Environment Analysis**
  - Firmware, developer, backend server, etc.
- **Connections**
  - GRPS, Wi-Fi, IRDA, Bluetooth, or NFC
- **Sensitive Data**
  - UDID, MAC, IMEI, etc.
- **Application Own Data**
  - XML, PList, SQLite, etc.
- **Application Structure**
  - Design, implementation

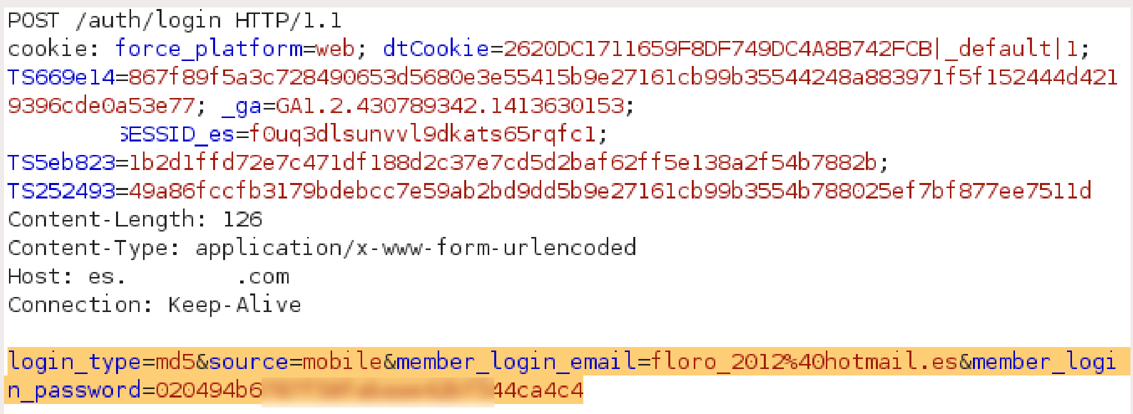
	Environment Analysis	Connections	Sensitive Data	Application Own Data	Application Structure
M1	√	√			
M2				√	√
M3		√			√
M4		√	√	√	√
M5	√			√	√
M6				√	√
M7	√	√			√
M8					√
M9				√	√
M10	√				√

Motivation

- **Software systems are prone to contain vulnerabilities** [1, 2]
  - Exploitable vulnerabilities: Buffer overflows, XSS, SQL injection, etc.
- Vulnerability detection: **Expensive and burdensome** [3]
- Mobile app vulnerabilities may put in risk user privacy data
- We propose an auditing methodology for mobile apps:
  - Focused on **OWASP 2014 Top Ten Mobile Risk**s [4]
  - **Five analysis blocks**, covering all aspects
  - **Case study: eCommerce app** available at app markets
    - **Vulnerabilities found** in backend servers and the app itself
- Our aim: **To guide an auditor when analysing a mobile app**

Case Study: eCommerce app (Android and iOS platforms)

- Spanish outlet app. **Full report available at [5]**
- **On Pre-Runtime Phase**
  - CVE-2004-0230 vulnerability in server side [6]
  - Weak ciphers in server side (RC4-MD5 and RC4-SHA)
  - Login passwords stored in MD5
  - Login username (i.e., email) in plain text (SQLite3 file)
  - Excess of permissions in both platforms
- **On Runtime Phase**
  - Login requests (plain-text username, MD5 password) sent via HTTP

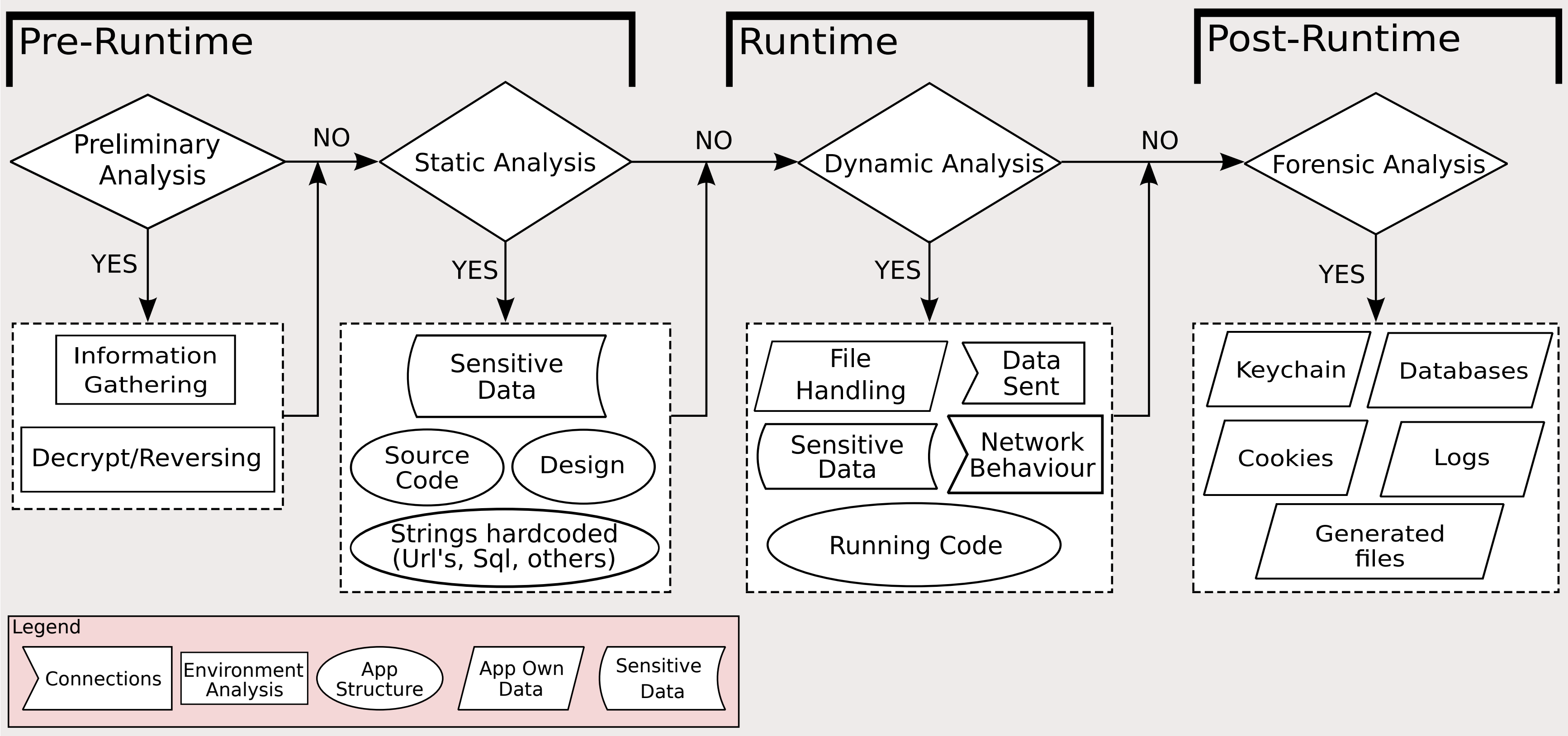


```
POST /auth/login HTTP/1.1
cookie: force_platform=web; dtCookie=2620DC1711659F8DF749DC4A8B742FCB|_default|1;
TS669e14=867f89f5a3c728490653d5680e3e55415b9e27161cb99b35544248a883971f5f152444d421
9396cde0a53e77; _ga=GA1.2.430789342.1413630153;
         SESSID_es=f0uq3dlsunvvl9dkats65rqfc1;
TS5eb823=1b2d1ffd72e7c471df188d2c37e7cd5d2baf62ff5e138a2f54b7882b;
TS252493=49a86fccfb3179bdebcc7e59ab2bd9dd5b9e27161cb99b3554b788025ef7bf877ee7511d
Content-Length: 126
Content-Type: application/x-www-form-urlencoded
Host: es.        .com
Connection: Keep-Alive

login_type=md5&source=mobile&member_login_email=floro_2012%40hotmail.es&member_logi
n_password=020494b6[redacted]44ca4c4
```

  - Market retargeting sending relevant user data
  - Purchase requests sent via HTTPS but without certificate pinning
    - Card number, issuer, holder, expiration, and CVV code
- **On Post-Runtime Phase**
  - Files contain login credentials in plain text
    - **Android**: Email in plain text, password in MD5
    - **iOS**: Emai and password in plain text
  - Cookies stored in plain text

Methodology



Conclusions

- **Complete security auditing methodology developed**
- **Five analysis blocks defined over OWASP 2014 Top Ten Mobile Risks**
- Allows to **find vulnerabilities** and to **detect suspicious behaviours**
- **Validated through a real case study, finding several vulnerabilities**
  - Spanish outlet app

References

1. **Charette, R.**: *Why software fails [software failure]*. IEEE Spectrum 42(9) (September 2005) 42–49
2. **Jimenez, W. et al.**: *Software Vulnerabilities Prevention and Detection Methods: A Review*. In: Procs. SEC-MDA 2009, CTIT Workshop Procs. Series (Jun 2009) 6–13
3. **Dowd, M. et al.**: *The Art of Software Security Assessment: Identifying and Preventing Software Vulnerabilities*. Addison-Wesley (2006)
4. **OWASP 2014 Top Ten Mobile Risks**: `https://www.owasp.org/index.php/OWASP_Mobile_Security_Project`
5. **CVE-2004-0230 vulnerability details**: `http://www.cvedetails.com/cve/CVE-2004-0230`
6. **Security Audit Report of eCommerce App**: `http://grupos.unileon.es/riasc/files/2015/01/eCommerceAuditReport.pdf`

Contact Information

- **Corresponding author**: alvaro.botas@unileon.es